ECE 375 PRELAB FOR LAB 5

Large Number Arithmetic

Lab Time: Friday 4 - 6

Faaiq Waqar

PreLab Questions

To complete this prelab, you may find it useful to look at the AVR Starter Guide and the AVR Instruction Set Manual. If you consult any online sources to help answer the pre lab questions, you must list them as references in your prelab.

1. For this lab, you will be asked to perform arithmetic operations on numbers that are larger than 8 bits. To be successful at this, you will need to understand and utilize many of the various arithmetic operations supported by the AVR 8-bit instruction set. List and describe all of the addition, subtraction, and multiplication instructions (i.e. ADC, SUBI, FMUL, etc.) available in AVR's 8-bit instruction set.

ADD - Add Two Registers: Add the components of 2 registers, a specified source register and a specified destination register, either of which can be any of the 31 registers from the cpu register file

ADC - Add with Carry of Two Registers: Add the components of 2 registers as well as the bit contained in the carry bit in the status register, which is typically carried over from a previous mathematical operation, a specified source register and a specified destination register, either of which can be any of the 31 registers from the cpu register file.

ADIW - Add Immediate to Word: Add an immediate value K ranging from value 0 to 63 to a register pair,a nd places the result in the register pair. This is a primary use case for the address registers, which need the word pair for addressing operation manipulation.

SUB - Subtract with two registers: Subtract the components of 2 registers, a specified source register and a specified destination register, either of which can be any of the 31 registers from the cpu register file

SUBI - Subtract constant from register: Subtract a constant K of value 0 to 255 from a specified destination register, this being registers 16 to 31 in the CPU register file

SBC - Subtract with carry two registers: Subtract the components of 2 registers as well as the included carry bit from the status register, a specified source register and a specified destination register, either of which can be any of the 31 registers from the cpu register file

SBCI - Subtract Immediate with Carry Set Bit in i/O Register: Subtracts a constant from a register and subtracts using the carry flag from the status register, and sets into destination register from r16-30

SBIW - Subtract immediate from word: Subtracts an immediate value K ranging from value 0 to 63 to a register pair,and places the result in the register pair. This is a primary use case for the address registers, which need the word pair for addressing operation manipulation.

MUL - Multiply Unsigned: Takes 2 unsigned multiplicands and multipliers, multiplies their 8 bit values and then places the word pair into the registers r1:r0 for product high and product low, can be used with registers for source and destination 16-31

MULS - Multiply Signed: Takes 2 signed multiplicands and multipliers, multiplies their 8 bit values and then places the word pair into the registers r1:r0 for product high and product low, can be used with registers for source and destination 16-31

MULSU - Multiply signed with unsigned: Takes signed and unsigned multiplicands and multipliers, multiplies their 8 bit values and then places the word pair into the registers r1:r0 for product high and product low, can be used with registers for source and destination 16-31

FMUL - Fractional Multiply Signed: Takes 2 fractional unsigned multiplicands and multipliers, multiplies their 8 bit values and then places the word pair into the registers r1:r0 for product high and product low, can be used with registers for source and destination 16-23

FMULS - Fractional Multiply Unsigned: Takes 2 fractional signed multiplicands and multipliers, multiplies their 8 bit values and then places the word pair into the registers r1:r0 for product high and product low, can be used with registers for source and destination 16-23

FMULSU - Fractional Multiply signed with Unsigned: Takes 1 fractional unsigned and 1 fractional signed multiplicands and multipliers, multiplies their 8 bit values and then places the word pair into the registers r1:r0 for product high and product low, can be used with registers for source and destination 16-23

2. Write pseudocode for an 8-bit AVR function that will take two 16-bit numbers (from data memory addresses $0111:$0110 and $0121:$0120), add them together, and then store the 16-bit result (in data memory addresses $0101:$0100). (Note: The syntax "$0111:$0110" is meant to specify that the function will expect little-endian data, where the highest byte of a multi-byte value is stored in the highest address of its range of addresses.)

; Set the Z register to the value of memory $0100

; Set the X Register to Value of $0110

; Set the Y Register to Value of $0120

; Load Value of X in R16, Post Increment X

; Load Value of Y in R17, Post Increment Y

; Add Value of R16 from R17, store in R16

; Store Value of R16 in Z, Post Increment Z

; Load Value of X in R16

; Load Value of Y in R17

; Add with Carry R16 from R17, place into R16

; Load Value of R16 into Z

3. Write pseudocode for an 8-bit AVR function that will take the 16-bit number in $0111:$0110, subtract it from the 16-bit number in $0121:$0120, and then store the 16-bit result into $0101:$0100.

; Set the Z register to the value of memory $0101

; Set the X Register to Value of $0111

; Set the Y Register to Value of $0121

; Load Value of X in R16, Post Decrement X

; Load Value of Y in R17, Post Decrement Y

; Subtract Value of R16 from R17, store in R16

; Store Value of R16 in Z, Post Decrement Z

; Load Value of X in R16

; Load Value of Y in R17

; Subtract with Carry R16 from R17, place into R16

; Load Value of R16 into Z